Non-negative Spectral Learning for Linear Sequential Systems

Hadrien Glaude, Cyrille Enderli, Olivier Pietquin

Non-negative Spectral Learning for Linear Sequential Systems

Hadrien Glaude[1,2], Cyrille Enderli[1], and Olivier Pietquin[2,3]

[1] Thales Airborne Systems, Elancourt, France
[2] Univ. Lille, CRIStAL, UMR 9189, SequeL Team, Villeneuve d'Ascq, France
[3] Institut Universitaire de France (IUF)

Abstract. Method of moments (MoM) has recently become an appealing alternative to standard iterative approaches like Expectation Maximization (EM) to learn latent variable models. In addition, MoM-based algorithms come with global convergence guarantees in the form of finite sample bounds. However, given enough computation time, by using restarts and heuristics to avoid local optima, iterative approaches often achieve better performance. We believe that this performance gap is in part due to the fact that MoM-based algorithms can output negative probabilities. By constraining the search space, we propose a non-negative spectral algorithm (NNSpectral) avoiding computing negative probabilities by design. NNSpectral is compared to other MoM-based algorithms and EM on synthetic problems of the PAutomaC challenge. Not only, NNSpectral outperforms other MoM-based algorithms, but also, achieves very competitive results in comparison to EM.

1 Introduction

Traditionally, complex probability distributions over structured data are learnt through generative models with latent variables (*e.g* Hidden Markov Models (HMM)). Maximizing the likelihood is a widely used approach to fit a model to the gathered observations. However, for many complex models, the likelihood is not convex. Thus, algorithms such as Expectation Maximization (EM) and gradient descent, which are iterative procedures, converge to local minima. In addition to being prone to get stuck into local optima, these algorithms are computationally expensive, to the point where obtaining good solutions for large models becomes intractable. A recent alternative line of work consists in designing learning algorithms for latent variable models exploiting the so-called Method of Moments (MoM). The MoM leverages the fact that low order moments of distributions contain most of the distribution information and are typically easy to estimate. The MoM have several pros over iterative methods. It can provide extremely fast learning algorithms as estimated moments can be computed in a time linear in the number of samples. MoM-based algorithms are often consistent with theoretical guarantees in form of finite-sample bounds. In addition, these algorithms are able to learn a large variety of models [1], [3], [15], [11]. In a recent work, [15] showed that numerous models are encompassed into the common framework of linear Sequential Systems (SSs) or equivalently Multiplicity Automata (MA). Linear SSs represent real functions over a set of words $f_{\mathcal{M}} : \Sigma^\star \to \mathbb{R}$, where Σ is an alphabet. In particular, they

can be used to represent probability distributions over sequences of symbols. Although, for the sake of clarity, we focus on learning stochastic rational languages (defined in Section 2), our work naturally extends to other equivalent or encompassed models: Predictive State Representations (PSRs), Observable Operator Models (OOMs), Partially Observable Markov Decision Processes (POMDPs) and HMMs.

Beyond all the appealing traits of MoM-based algorithms, a well-known concern is the negative-probabilities problem. Actually, most of MoM-based algorithms do not constrain the learnt function to be a distribution. For some applications requiring to learn actual probabilities, this is a critical issue. For example, a negative and unnormalized measure does not make sense when computing expectations. Sometimes, an approximation of a probability distribution is enough. For instance, computing the Maximum a Posteriori (MAP) only requires the maximum to be correctly identified. But even for these applications, we will show that constraining the learned function to output non-negative values helps the learning process and improves the model accuracy. A second concern is the observed performance gap with iterative algorithms. Although they usually need restarts and other heuristics to avoid local minima, given enough time to explore the space of parameters, they yield to very competitive models in practice. Recently, an empirical comparison [5] have shown that MoM-based algorithms still perform poorly in comparison to EM.

In this paper, we propose a new MoM-based algorithm, called non-negative spectral learning (NNSpectral) that constraints the output function to be non-negative. Inspired by theoretical results on MA defined on non-negative commutative semi-rings, NNSpectral uses Non-negative Matrix Factorization (NMF) and Non-Negative Least Squares (NNLS). An empirical evaluation on the same twelve synthetic problems of the PAutomaC challenge used by [5] allows us fairly comparing NNSpectral to three other MoM-based algorithms and EM. Not only, NNSpectral outperforms previous MoM-based algorithms, but also, it is the first time to our knowledge that a MoM-based algorithm achieves competitive results in comparison to EM.

2 Multiplicity Automata

2.1 Definition

Let Σ be a set of symbols, also called an alphabet. We denote by $\Sigma^\star$, the set of all finite words made of symbols of Σ, including the empty word ε. Words of length k form the set Σ^k. Let u and $v \in \Sigma^\star$, uv is the concatenation of the two words and $u\Sigma^\star$ is the set of finite words starting by u. In the sequel, K is a commutative semi-ring, in particular $\mathbb{R}$ or $\mathbb{R}^+$. We are interested in mapping of $\Sigma^\star$ into K called formal power series. Let f be a formal power series, for a set of words $\mathcal{S}$, we define $f(\mathcal{S}) = \sum_{u \in \mathcal{S}} f(u) \in K$. Some formal power series can be represented by compact models, called MA.

Definition 1 (Multiplicity Automaton). *Let K be a semi-ring, a K-multiplicity automaton (K-MA) with n states is a structure $\left\langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty \right\rangle$, where Σ is an alphabet and Q is a finite set of states. Matrices $A_o \in K^{n\times n}$ contain the transition weights. The vectors $\boldsymbol{\alpha}_\infty \in K^n$ and $\boldsymbol{\alpha}_0 \in K^n$ contain respectively the terminal and*

initial weights. A K-MA $\mathcal{M}$ defines a rational language $r_{\mathcal{M}} : \Sigma^\star \to \mathbb{R}$,

$$r_{\mathcal{M}}(u) = r_{\mathcal{M}}(o_1 \dots o_k) = \boldsymbol{\alpha}_0^\top A_u \boldsymbol{\alpha}_0 = \boldsymbol{\alpha}_0^\top A_{o_1} \dots A_{o_k} \boldsymbol{\alpha}_\infty.$$

A function f is said *realized* by a K-MA $\mathcal{M}$, if $r_{\mathcal{M}} = f$.

Definition 2 (Rational language). *A formal power series r is rational over K iff it is realized by a K-MA.*

Two K-MA that define the same rational language are *equivalent*. A K-MA is *minimal* if there is not an equivalent K-MA with strictly fewer states. An important operation on K-MA is the *conjugation* by an invertible matrix. Let $R \in K^{n \times n}$ be invertible and $\mathcal{M} = \left\langle \Sigma, Q, \{A_o\}_{o \in \Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty \right\rangle$ be a K-MA of dimension n, then

$$\mathcal{M}' = \left\langle \Sigma, Q, \left\{R^{-1} A_o R\right\}_{o \in \Sigma}, R^\top \boldsymbol{\alpha}_0, R^{-1} \boldsymbol{\alpha}_\infty \right\rangle,$$

defines a conjugated K-MA. In addition, $\mathcal{M}$ and $\mathcal{M}'$ are equivalent. In this paper, we are interested in learning languages representing distributions over finite-length words that can be compactly represented by a K-MA. When $K = \mathbb{R}$, this forms the class of stochastic rational language.

Definition 3 (Stochastic rational language). *A stochastic rational language p is a rational language with values in $\mathbb{R}^+$ such that $\sum_{u \in \Sigma^\star} p(u) = 1$.*

Stochastic rational languages are associated to the following subclass of $\mathbb{R}$-MA.

Definition 4 (Stochastic Multiplicity Automaton). *A stochastic multiplicity automaton (SMA) $\mathcal{M}$ is a $\mathbb{R}$-MA realizing a rational stochastic language.*

2.2 Hankel Matrix Representation

Let $f : \Sigma^\star \to K$ be a formal power series, we define $H_f \in K^{\Sigma^\star \times \Sigma^\star}$ the bi-infinite Hankel matrix whose rows and columns are indexed by $\Sigma^\star$ such that $H_f[u, v] = f(uv)$,

$$H_f = \begin{array}{c} \\ \varepsilon \\ a \\ b \\ aa \\ \vdots \end{array} \begin{array}{c} \begin{array}{ccccc} \varepsilon & a & b & aa & \dots \end{array} \\ \begin{pmatrix} f(\varepsilon) & f(a) & f(b) & f(aa) & \dots \\ f(a) & f(aa) & f(ab) & f(aaa) & \dots \\ f(b) & f(ba) & f(bb) & f(baa) & \dots \\ f(aa) & f(aaa) & f(aab) & f(aaaa) & \dots \\ \vdots & \vdots & \vdots & \vdots & \ddots \end{pmatrix} \end{array}$$

When f is a stochastic language, H_f contains occurring probabilities that can be estimated from samples by empirical counts. Details about matrices defined over semirings can be found in [12]. We note H the Hankel matrix when its formal power series can be inferred from the context. Let for all $o \in \Sigma$, $H_o \in K^{\Sigma^\star \times \Sigma^\star}$, $\mathbf{h}_S \in K^{\Sigma^\star}$ and $\mathbf{h}_P \in K^{\Sigma^\star}$ be such that $H_o(u, v) = f(uov)$, $\mathbf{h}_S(u) = \mathbf{h}_P(u) = f(u)$. These vectors and matrices can be extracted from H. The Hankel representation of formal series lies in the heart of all MoM-based learning algorithms, because of the following fundamental theorem.

Theorem 1 (See [7]). *Let r be a rational language over K and $\mathcal{M}$ a K-MA with n states that realizes it, then $\mathrm{rank}(H_r) \leq n$. Conversely, if the Hankel matrix H_r of a formal power series r has a finite rank n, then r is a rational language over K and can be realized by a minimal K-MA with exactly n states.*

Note that, the original proof assumes that K is a field but remains true when K is a commutative semi-ring, as ranks, determinants and inverses are well defined in semi-modules. In addition, the proof gives also the construction of H from a K-MA and *vice-versa*. For a K-MA with n states, observe that $H[u,v] = (\boldsymbol{\alpha}_0^\top A_u)(A_v \boldsymbol{\alpha}_\infty)$. Let $P \in K^{\Sigma^\star \times n}$ and $S \in K^{n \times \Sigma^\star}$ be matrices defined as follows,

$$P = ((\boldsymbol{\alpha}_0^\top A_u)^\top)_{u \in \Sigma^\star}^\top, \qquad S = (A_v \boldsymbol{\alpha}_\infty)_{v \in \Sigma^\star},$$

then $H = PS$. Moreover, we have that,

$$H_o = P A_o S, \quad \mathbf{h}_S^\top = \boldsymbol{\alpha}_0^\top S, \quad \mathbf{h}_P = P \boldsymbol{\alpha}_\infty. \tag{1}$$

So the K-MA parameters can be recovered by solving eq. (1). Hopefully, we do not need to consider the bi-infinite Hankel matrix to recover the underlying K-MA. Given a basis $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ of prefixes and suffixes, we denote by $H_\mathcal{B}$ the sub-block of H. A basis $\mathcal{B}$ is *complete* if $H_\mathcal{B}$ has the same rank than H. A basis is *suffix-closed* if $\forall u \in \Sigma^\star$, $\forall o \in \Sigma$, $ou \in \mathcal{S} \Rightarrow u \in \mathcal{S}$. In [4], the author shows that if $\mathcal{B} = (\mathcal{P}, \mathcal{S})$ is a suffix-closed complete basis, by defining P over $\mathcal{P}$, S over $\mathcal{S}$ and H over $\mathcal{B}$, we can recover a MA using eq. (1).

2.3 Spectral Learning

This section reviews the Spectral Learning algorithm to learn $\mathbb{R}$-MA from samples generated by a stochastic rational language. In the literature, severals methods are used to build a suffix-closed basis from data. For example, one can use all prefixes and suffixes that appear in the training set. In addition, we require that sets of prefixes and suffixes contain the empty word ε. Once a basis is chosen, the Spectral algorithm first estimates the probabilities in $H_\mathcal{B}$ by empirical counts. Then, it recovers a factorized form of $H_\mathcal{B} = UDV^\top$ through a truncated Singular Value Decomposition (SVD). Finally, setting $P = UD$ and $S = V^\top$, the algorithm solves eq. (1). More precisely, let $\mathbf{1}_\varepsilon^S$ and $\mathbf{1}_\varepsilon^P$ be vectors filled with 0s with a single 1 at the index of the empty word in the basis, we have that $\mathbf{h}_S = (\mathbf{1}_\varepsilon^P)^\top H_\mathcal{B}$, $\mathbf{h}_P = H_\mathcal{B} \mathbf{1}_\varepsilon^S$. Let, for all $o \in \Sigma^\star$, $T_o \in \mathbb{R}^{|\mathcal{S}| \times |\mathcal{S}|}$ be matrices such that, $T_o[u,v] = \delta_{u=ov}$ then we have that $H_o = H_\mathcal{B} T_o$. Using these identities in eq. (1), one obtains Algorithm 1. In the experiments, following the advices of [8], we normalized the feature-variance of the coefficients of the Hankel matrix by independently scaling each row and column by a factor $c_u = \sqrt{|\mathcal{S}| / (\#u + 5)}$, where $\#u$ is the number of occurrences of u. In addition, depending on the problem, it can be better to work with other series derived from p. For example, the substring-based series $p^{substring}(u) = \sum_{w,v \in \Sigma^\star} p(wuv)$ is related to p^{string}. According to [4], if p^{string} is realized by a SMA, then $p^{substring}$ is too. In addition, he provides an explicit conversion between string-based SMA and substring-based SMA preserving the number of states. For all algorithms compared in Section 4, we used the series leading to the best results for each problem.

Algorithm 1 Spectral algorithm for $\mathbb{R}$-MA.

1: Choose a set of prefixes $\mathcal{P} \subset \Sigma^\star$ and suffixes $\mathcal{S} \subset \Sigma^\star$ both containing ε
2: Using $\mathcal{S}$, build the matrices T_o for all $o \in \Sigma$ such that $T_o[u, v] = \delta_{u=ov}$
3: Estimate $H_\mathcal{B}$ by empirical counts.
4: $U, D, V = \mathcal{SVD}_n(H_\mathcal{B})$ using the truncated SVD, where n is a parameter of the algorithm
5: For all $o \in \Sigma$ do $A_o = V^\top T_o V$
6: $\boldsymbol{\alpha}_0^\top = (\mathbf{1}_\varepsilon^P)^\top H_\mathcal{B} V$
7: $\boldsymbol{\alpha}_\infty = V^\top \mathbf{1}_\varepsilon^S$

3 Non-negative Spectral Learning

As mentioned in the introduction, Algorithm 1 is designed to learn $\mathbb{R}$-MA and is very unlikely to return a SMA. This is a major drawback when a probability distribution is required. In this case, one has to rely on heuristics like thresholding the values to be contained in $[0, 1]$, which introduces errors in predictions. A natural enhancement of the Spectral algorithm would be to constraint the return model to be a SMA. Unfortunately, this is not likely to be feasible due to the underlying complexity between $\mathbb{R}$-MA and SMA. Indeed, although $\mathbb{R}$-MA are strictly more general than SMA, checking whether a $\mathbb{R}$-MA is stochastic is undecidable [9]. In terms of algorithms, constraining the return model to be a SMA would require adding an infinite number of constraints. As a matter of fact, a $\mathbb{R}$-MA realizing a language r requires the non-negativity ($\forall u \in \Sigma^\star$, $r(x) \geq 0$) and the convergence to 1 ($\lim_{L\to+\infty} \sum_{l=0}^{L} \sum_{u\in\Sigma^l} r(u) = 1$) of the series to be a SMA. The undecidability comes only from the non-negativeness. Note that only the existence of the limit is really required as a convergent series can always be normalized. Thus, we propose to restrict learning to $\mathbb{R}^+$-MA, which by definition produces non-negative series. Although, SMA are not included in $\mathbb{R}^+$-MA, there are $\mathbb{R}^+$-MA realizing probability distributions and called Probabilistic Finite Automata (PFA). Relations between all these classes of MA are summed up in Figure 1.

Definition 5 (Probabilistic (Deterministic) Finite Automaton). *A probabilistic finite automaton (PFA) $\mathcal{M} = \langle \Sigma, Q, \{A_o\}_{o\in\Sigma}, \boldsymbol{\alpha}_0, \boldsymbol{\alpha}_\infty \rangle$ is a SMA with non-negative weights verifying $\mathbf{1}^\top \boldsymbol{\alpha}_0 = 1$, $\boldsymbol{\alpha}_\infty + \sum_{o\in\Sigma} A_o \mathbf{1} = \mathbf{1}$. The weights of PFA are in $[0, 1]$ and can be viewed as probabilities over transitions, initial states and terminal states. A Probabilistic Deterministic Infinite Automata (PDFA) if a PFA with deterministic transitions.*

Thus, in contrast to the Spectral algorithm which returns a $\mathbb{R}$-MA, NNSpectral returns a $\mathbb{R}^+$-MA avoiding the non-negativity probability problem. The NNSpectral algorithm, given in Algorithm 2, also uses the decomposition in Theorem 1, but applied to MA defined on the semi-ring $\mathbb{R}^+$ instead of the field $\mathbb{R}$. Thus, there exists a low-rank factorization with non-negative factors of the Hankel matrix representing a $\mathbb{R}^+$-MA. Finding such a decomposition is a well-known problem, called NMF. It aims at finding a low rank decomposition of a given non-negative data matrix $H_\mathcal{B} \in \mathbb{R}^{n\times m}$ such that $\|H_\mathcal{B} - PS\|_F$ where $P \in \mathbb{R}^{n\times r}$ and $S \in \mathbb{R}^{r\times m}$ are component-wise non-negative and $\|\cdot\|_F$ is the Frobenius norm. Unfortunately, NMF is NP-Hard in general [16] and the decomposition is not unique which makes the problem ill-posed. Hence, in practice, heuristic algorithms are used which have only the guarantee to converge to a stationary

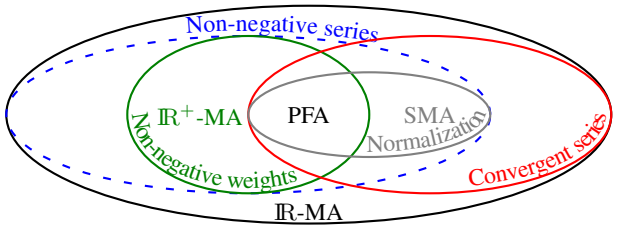

Fig. 1. Solid lines are polynomially decidable. Dashed line are undecidable. See [2][9].

point. Most of them run in $\mathcal{O}(nmr)$. We refer the reader to [10] for a comprehensive survey of these algorithms. In the experiments, the alternating non-negative least squares algorithm has shown to be a good trade-off between convergence speed and quality of the approximation. This algorithm iteratively optimizes P and S by solving NNLS problems. NNLS is a non-negative constrained version of the least squares method. Being equivalent to a quadratic programming problem under linear constraints, it is convex and algorithms converge to the optimum. In the experiments, NNLS problems were solved using the projected gradient algorithm of [14]. So, given a non-negative factorization of $H_{\mathcal{B}}$, solving eq. (1) is done by NNLS to ensure the non-negativity of the weights. Although NNSpectral cannot learn SMA (or the equivalent OOMs, PSRs), a SMA can be arbitrarily well approximated by a PFA (or the equivalent HMMs, POMDPs) with a growing number of states [13].

Algorithm 2 NNSpectral algorithm for $\mathbb{R}^+$-MA

1: Choose a set of prefixes $\mathcal{P} \subset \Sigma^\star$ and suffixes $\mathcal{S} \subset \Sigma^\star$ both containing ε
2: Using $\mathcal{S}$, build the matrices T_o for all $o \in \Sigma$ such that $T_o[u, v] = \delta_{u=ov}$
3: Estimate $H_{\mathcal{B}}$ by empirical counts.
4: $P, S \leftarrow \operatorname{argmin}_{PS} \|H_{\mathcal{B}} - PS\|_F$ s.t. $P \in \mathbb{R}^{|\mathcal{P}| \times n}, S \in \mathbb{R}^{n \times |\mathcal{S}|} \geq 0$ ▷ by NMF
5: For all $o \in \Sigma$ do $A_o \leftarrow \operatorname{argmin}_{A \geq 0} \|AS - ST_o\|_F$ ▷ by NNLS
6: $\boldsymbol{\alpha}_0 \leftarrow \operatorname{argmin}_{\boldsymbol{\alpha} \geq 0} \|\boldsymbol{\alpha}^\top S - (\mathbf{1}_\varepsilon^P)^\top H_{\mathcal{B}}\|_F$ ▷ by NNLS
7: $\boldsymbol{\alpha}_\infty \leftarrow S\mathbf{1}_\varepsilon^S$

4 Numerical Experiments

The Probabilistic Automata learning Competition (PAutomaC) is dealing with the problem of learning probabilistic distributions from strings drawn from finite-state automata. From the 48 problems available, we have selected the same twelve problems than in [5], to provide a fair comparison with other algorithms. The generating model can be of three kinds: PFA, HMMs or PDFA. Four models have been selected from each class. A detailed description of each problem can be found in [17]. Table 1 compares the best results of NNSpectral between learning from strings or substrings to EM and the best

	Perplexity				WER			
ID	NNSpectral	EM	MoM	True model	NNSpectral	EM	MoM	True model
HMM 1	**30.54**(64)	500.10	44.77	29.90(63)	72.7(30)	75.7	**71.3**	68.8(63)
14	116.98(11)	**116.84**	128.53	116.79(15)	**68.8**(7)	68.8	70.0	68.4(15)
33	32.21(14)	**32.14**	49.22	31.87(13)	**74.3**(6)	74.3	76.7	74.1(13)
45	**24.08**(2)	107.75	31.87	24.04(14)	78.24(2)	**78.1**	80.1	78.1(14)
PDFA 6	76.99(28)	**67.32**	95.12	66.98(19)	**47.1**(21)	47.4	50.2	46.9(19)
7	**51.26**(12)	51.27	62.74	51.22(12)	48.41(42)	**48.1**	50.6	48.3(12)
27	**43.81**(46)	94.40	102.85	42.43(19)	**73.9**(20)	83.0	75.5	73.0(19)
42	**16.12**(20)	168.52	23.91	16.00(6)	**56.6**(8)	58.1	61.4	56.6(6)
PFA 29	25.24(35)	**25.09**	34.57	24.03(36)	47.6(36)	49.2	**47.3**	47.2(36)
39	**10.00**(6)	10.43	11.24	10.00(6)	**59.4**(19)	63.3	62.0	59.3(6)
43	**32.85**(7)	461.23	36.61	32.64(67)	**76.8**(25)	77.4	78.0	77.1(67)
46	12.28(44)	**12.02**	25.28	11.98(19)	78.0(20)	**77.5**	79.4	77.3(19)

Table 1. Comparison with other algorithm for Perplexity (left table) and WER (right table). "MoM" stands for the best of MoM-based algorithms. Model sizes are listed in parentheses.

results among the following MoM-based algorithms : CO [6] using strings, Tensor [1] using strings and Spectral using strings and substrings. A description and comparison of these algorithms can be found in [5].

The quality of a model can be measured by the quality of the probability distribution it realizes. The objective is to learn a MA realizing a series p close to the distribution $p_\star$, which generated the training set $\mathcal{T}$. The quality of p is measured by the perplexity that corresponds to the average number of bits needed to represent a word using the optimal code given by $p_\star$.

$$\text{Perplexity}(\mathcal{M}) = 2^{-\sum_{u\in\mathcal{T}} p_\star(u)\log(p_\mathcal{M}(u))}$$

The quality of model can also be evaluated by the Word Error Rate (WER) metric. It measures the fraction of incorrectly one-step-ahead predicted symbols.

In the simulations, we perform a grid search to find the optimal rank for each of the performance metric. For each problem, the best algorithm is indicated by a bold number and the best score between NNSpectral and other MoM-based algorithms is underlined. The score of the true model is reported for comparison. For the perplexity, NNSpectral outperforms other MoM-based algorithm on the 12 problems and does better that EM for 7 problems. For the other 5 problems, NNSpectral achieves performances very close to EM and to the true model. For the WER metric, NNSpectral beats other MoM-based algorithms on 10 problems and scores at top on 7 problems. Also, for all problems NNSpectral produces perplexity close to the optimal ones, whereas, on 5 problems EM fails to produce an acceptable solution.

5 Conclusion

In this paper, we proposed a new algorithm inspired by the theoretical developments of MA defined on semi-rings. NNSpectral works by constraining the search space

to ensure non-negativity of the rational language. Like other MoM-based algorithms, NNSpectral is able to handle large-scale problems much faster than EM. Here, the consistency is lost due to the use of heuristics to solve the NMF problem. Experimentally, this does not seem to be a major problem because NNSpectral outperforms other MoM-based algorithms and is competitive with EM that sometimes produces aberrant solutions. Thus, NNSpectral provides a good alternative to the EM algorithm with a much lower computational cost. In further works, we would like to investigate how using NNSpectral to initialize an EM algorithm and how adding constraints in NNSpectral to ensure the convergence of the series. In addition, relations with NMF could be further exploit to produce tensor [1] or kernel-based algorithms.

References

1. Anandkumar, A., Ge, R., Hsu, D., Kakade, S.M., Telgarsky, M.: Tensor decompositions for learning latent variable models. arXiv preprint arXiv:1210.7559 (2012)
2. Bailly, R., Denis, F.: Absolute convergence of rational series is semi-decidable. Information and Computation 209(3), 280–295 (2011)
3. Bailly, R., Habrard, A., Denis, F.: A spectral approach for probabilistic grammatical inference on trees. In: Proc of ALT-10. pp. 74–88. Springer (2010)
4. Balle, B.: Learning finite-state machines: algorithmic and statistical aspects. Ph.D. thesis (2013)
5. Balle, B., Hamilton, W., Pineau, J.: Methods of moments for learning stochastic languages: Unified presentation and empirical comparison. In: Proc. of ICML-14. pp. 1386–1394 (2014)
6. Balle, B., Quattoni, A., Carreras, X.: Local loss optimization in operator models: A new insight into spectral learning. In: Proc. of ICML-12 (2012)
7. Carlyle, J.W., Paz, A.: Realizations by stochastic finite automata. Journal of Computer and System Sciences 5(1), 26 – 40 (1971)
8. Cohen, S.B., Stratos, K., Collins, M., Foster, D.P., Ungar, L.H.: Experiments with spectral learning of latent-variable pcfgs. In: Proc of HLT-NAACL-13. pp. 148–157 (2013)
9. Denis, F., Esposito, Y.: On rational stochastic languages. Fundamenta Informaticae 86(1), 41–77 (2008)
10. Gillis, N.: The Why and How of Nonnegative Matrix Factorization. ArXiv e-prints (Jan 2014)
11. Glaude, H., Pietquin, O., Enderli, C.: Subspace identification for predictive state representation by nuclear norm minimization. In: Proc. of ADPRL-14 (2014)
12. Guterman, A.E.: Rank and determinant functions for matrices over semirings. In: Young, N., Choi, Y. (eds.) Surveys in Contemporary Mathematics, pp. 1–33. Cambridge University Press (2007), cambridge Books Online
13. Gybels, M., Denis, F., Habrard, A.: Some improvements of the spectral learning approach for probabilistic grammatical inference. In: Proc. of ICGI-12. vol. 34, pp. 64–78 (2014)
14. Lin, C.J.: Projected gradient methods for nonnegative matrix factorization. Neural computation 19(10), 2756–2779 (2007)
15. Thon, M., Jaeger, H.: Links between multiplicity automata, observable operator models and predictive state representationsa unified learning framework. Journal of Machine Learning Research (to appear)
16. Vavasis, S.A.: On the complexity of nonnegative matrix factorization. SIAM Journal on Optimization 20(3), 1364–1377 (2009)
17. Verwer, S., Eyraud, R., de la Higuera, C.: Results of the pautomac probabilistic automaton learning competition. Journal of Machine Learning Research - Proceedings Track 21, 243–248 (2012)